Fisher Mask Nodes for Language Model Merging

Thennal D K, Ganesh Nathan, Suchithra M S

Indian Institute of Information Technology Kottayam
Kerala, India
{thennal21bcs14, ganesh21bcs104, suchithra}@iiitkottayam.ac.in

Abstract

Fine-tuning pre-trained models provides significant advantages in downstream performance. The ubiquitous nature of pre-trained models such as BERT and its derivatives in natural language processing has also led to a proliferation of task-specific fine-tuned models. As these models typically only perform one task well, additional training or ensembling is required in multi-task scenarios. The growing field of model merging provides a solution, dealing with the challenge of combining multiple task-specific models into a single multi-task model. In this study, we introduce a novel model merging method for Transformers, combining insights from previous work in Fisher-weighted averaging and the use of Fisher information in model pruning. Utilizing the Fisher information of mask nodes within the Transformer architecture, we devise a computationally efficient weighted-averaging scheme. Our method exhibits a regular and significant performance increase across various models in the BERT family, outperforming full-scale Fisher-weighted averaging in a fraction of the computational cost, with baseline performance improvements of up to +6.5 and a speedup between 57.4x and 321.7x across models. Our results prove the potential of our method in current multi-task learning environments and suggest its scalability and adaptability to new model architectures and learning scenarios.

Keywords: model merging, Fisher information, NLP, GLUE, transformers, BERT

1. Introduction

In recent years, pre-trained models have become ubiquitous in natural language processing (NLP) (Qiu et al., 2020; Han et al., 2021; Zhou et al., 2023; Min et al., 2021). In particular, models inheriting the Transformer architecture (Vaswani et al., 2017) such as BERT and its variants (Devlin et al., 2019) are popular for a wide variety of tasks (Zhou et al., 2023; Min et al., 2021). By finetuning on downstream tasks, pre-trained models achieve improved performance on those specific tasks. However, individual fine-tuned models for each task necessitates significant overhead in multi-task scenarios, as multiple separate models need to be running at once (Zhang and Yang, 2021; Fifty et al., 2021; Yadav et al., 2023). Furthermore, individual fine-tuned models sacrifice generalization capabilities (Jin et al., 2023; Arpit et al., 2022; Yadav et al., 2023; Ilharco et al., 2023). Fine-tuning on multiple tasks at once is a possible solution, but is generally resources-intensive and costly (Yadav et al., 2023; Fifty et al., 2021).

Model merging is a promising field that can alleviate this problem. Model merging predominantly deals with the challenges of fusing multiple separate models—generally of the same architecture but with different parameters—into a single model. A growing body of literature addresses the restricted context of merging multiple task-specific models into a single multitask model (Yadav et al., 2023; Matena and Raffel, 2022; Ilharco et al., 2023; Jin et al., 2023). Fisher-weighted averaging (Matena and Raffel, 2022) is one such method that uses the diagonal approximation of the Fisher information of each parameter as weights in a weighted-averaging scheme. However, doing so requires access to the validation set that the model will be evaluated on, along with significant resource overhead, as gradients for all parameters are required to calculate the approximated Fisher information.

Fisher information has been used extensively in machine learning as a measure of parameter importance (Kwon et al., 2022; Liu et al., 2021; Kirkpatrick et al., 2017a; Pascanu and Bengio, 2014; Soen and Sun, 2021). In recent years, it has gained particular prominence in the field of model pruning (Liu et al., 2021; Kwon et al., 2022). Kwon et al. (2022) devised a pruning scheme for Transformer models utilizing Fisher information matrices. Masks are inserted in every feed-forward layer and attention head, and the approximated Fisher information of the masks is used as an indication of the necessity of the affected block of parameters. Low-importance parameters, indicated by this metric, are then pruned.

Our work focuses on modifying the Fisher-weighted merging method devised by Matena and Raffel (2022) with the insights gained from Kwon et al. (2022): that the Fisher information of masks can act as an importance measure for their affected parameter blocks. We insert masks and calculate an approximate of the Fisher information of each mask. By associating the mask Fisher information to relevant parameters in an empirical

LREC-COLING 2024, pages 7349–7355

scheme, we merge models via weighted averaging similar to Matena and Raffel (2022). By only calculating the Fisher information of masks instead of the Fisher information of all parameters, our methods provide a massive reduction in computational cost while improving the performance benefits. As we restrict our context to performance on the respective tasks of fine-tuned model merges, our method does not require access to the validation set. The Fisher information matrix of each fine-tuned model on its respective task can provide a measure of how important each parameter is to that particular model, and we hypothesize that weighing by said Fisher information allows constructive merging, resulting in a single model performant in all given tasks.

We validate our hypothesis by demonstrating the effectiveness of our proposed methodology on various architectures of differing sizes, combining finetuned models on six tasks from the GLUE benchmark (Wang et al., 2018). Our method provides a significant improvement over simple averaging and Fisher-weighted merging in most models, using extremely small amounts of data and computational resources. We also note our method's efficiency at scale, with a speedup ranging from **57.4x** to **321.7x** across models in comparison to Fisher-weighted merging.

Our code is publicly released on Github.[1]

2. Related Work

Much of the work in model merging has been done in addressing permutation symmetry when models are trained from different initializations (Ainsworth et al., 2023; Singh and Jaggi, 2023; Jin et al., 2023; Li et al., 2016; Tatro et al., 2020; Entezari et al., 2022). However, in the case of different fine-tuned models trained from the same parent pre-trained model, merging can often be done directly. As the optimization trajectory overlaps at the beginning, even simple averaging can provide reasonable results (Choshen et al., 2022; Wortsman et al., 2022; Ilharco et al., 2023; Jin et al., 2023). In this setting, model merging has been explored for a wide variety of contexts including improving single-task performance (Choshen et al., 2022; Wortsman et al., 2022), transfer learning (Matena and Raffel, 2022), federated learning (Li et al., 2020; McMahan et al., 2016), and improving generalization (Jin et al., 2023; Ilharco et al., 2023; Arpit et al., 2022; Cha et al., 2021).

Restricting ourselves to the specific challenge of merging fine-tuned task-specific models for multi-task learning, many methods improve over the baseline of simple parameter averaging (Jin et al., 2023; Yadav et al., 2023; Ilharco et al., 2023; Matena and Raffel, 2022). Fisher-weighted averaging, proposed by Matena and Raffel (2022), was originally evaluated in the context of transfer learning but it has been further employed for multi-task learning (Yadav et al., 2023).

The Fisher information matrix as a statistical measure has been used extensively in machine learning for a wide variety of applications (Kirkpatrick et al., 2017a; Kwon et al., 2022; Pascanu and Bengio, 2014; Soen and Sun, 2021; Hannun et al., 2021; Liu et al., 2021). Liu et al. (2021) prominently uses it for the task of pruning across multiple model architectures. Kwon et al. (2022) provides a fast Transformer-specific pruning scheme using the Fisher information matrix of mask nodes.

As the full Fisher information matrix is infeasible to calculate for most applications, only the diagonals are computed over a specified sample as an approximation (Matena and Raffel, 2022; Kirkpatrick et al., 2017a). However, even the diagonal approximation incurs the same computational cost as training on a specified number of examples. Our method borrows the mask architecture from Kwon et al. (2022), applying it to the field of model merging to reduce computational cost while keeping the performance benefits of Fisher-weighted averaging. We also note that our method does not require access to the validation set, in contrast to other methods (Matena and Raffel, 2022; Yadav et al., 2023; Ilharco et al., 2023).

3. Methodology

Our approach relies on the hypothesis that the Fisher information of mask nodes can represent the importance of the parameters enclosed by the mask, as inspired by Kwon et al. (2022). Using the Fisher information of the mask nodes as a proxy for the Fisher information of the corresponding block of parameters, we merge parameters in a weighted-averaging merging scheme similar to Matena and Raffel (2022). We further consider only the training sets of the fine-tuned models for calculating the Fisher information in order to remove the dependency on the validation set and widen the applicability of our method.

In accordance with Kwon et al. (2022), our method focuses on the Transformer architecture, specifically the BERT architecture family (Devlin et al., 2019). BERT can be characterized as a stack of homogeneous Transformer encoder blocks, consisting of a multi-headed attention block followed by a feed-forward block. Masks are inserted on each head of the multi-headed attention block, and on each row in the intermediate linear layer of the feed-forward block, referred to as filters by Kwon et al. (2022).

Following the notation of the formal BERT algorithm given by Phuong and Hutter (2022), we rep-

[1] https://github.com/thennal10/fisher-nodes-merging

resent the modified multi-headed attention algorithm in Equation 1 and the modified feed-forward layer in Equation 2:

$$\begin{aligned}&\text{For } h \in \{1, ..., H\}:\\&\quad Y_h = \texttt{Attention}(X; W_{qkv}^{h,l})\\&Y = \{Y_1, Y_2, ..., Y_H\}\\&X \leftarrow X + W^o(Y \odot m_{mha}^l) + b^o\end{aligned} \tag{1}$$

$$\begin{aligned}X \leftarrow X + W_{mlp2}^l\texttt{GELU}(m_{mlp}^l \odot W_{mlp1}^l X + b_{mlp1}^l)\\+b_{mlp2}^l\end{aligned} \tag{2}$$

We assume a given input X and present only the modified section. We also omit the transposition of the bias vectors present in the original algorithm for conciseness.

In these equations, m_{mlp}^l and m_{mha}^l are mask vectors for the l-th layer of the feed-forward network and the multi-headed attention, respectively. $m_{mlp}^l = \mathbf{1}_{D\times 1}$ where D represents the output size of the intermediate layer (the number of rows of W_{mlp1}^l, or the number of filters as per Kwon et al. (2022)), and $m_{mha}^l = \mathbf{1}_{H\times 1}$ where H is the number of attention heads. The concatenated masks can thus be characterized as $m_{mha} = \mathbf{1}_{H\times L}$ and $m_{mlp} = \mathbf{1}_{R\times L}$, where L is the total number of layers. As the masks are only used for their representational value, their values stay constant and equal to 1 throughout the process.

The Fisher information matrix can be formulated in terms of the partial differentials of the loss function (Kwon et al., 2022):

$$I := \frac{1}{|D|}\sum_{(x,y)\in D}\left(\frac{\partial}{\partial m}L(x,y;1)\right)\left(\frac{\partial}{\partial m}L(x,y;1)\right)^T \tag{3}$$

As the full Fisher information matrix takes $O(|\theta|^2)$ to store, it becomes impractical for even a moderate number of masks. Thus, we use the diagonal approximation of the full matrix, as commonly practiced (Kirkpatrick et al., 2017b; Matena and Raffel, 2022; Kwon et al., 2022). The masks then have an associated Fisher information value equivalent to the mean squared sum of the gradients:

$$I_{ii} := \frac{1}{|D|}\sum_{(x,y)\in D}\left(\frac{\partial}{\partial m_i}L(x,y;1)\right)^2 \tag{4}$$

Matena and Raffel (2022) provide the following closed-form solution for model merging using the diagonal approximation of the Fisher information F_j for parameters θ_j of the j^{th} model, with M as the total number of models to be merged:

$$\theta^* = \frac{\sum_{j=1}^{M}\lambda_j F_j \theta_j}{\sum_{j=1}^{M}\lambda_j F_j} \tag{5}$$

We keep $\lambda_j = 1$ as we do not conduct hyperparameter tuning on the validation set. We approximate F_j of each parameter with the Fisher information I_{ii} of the mask node m_i according to the following scheme:

1. If the parameter is in the query and key matrices of a particular attention head, i.e. θ_j is in $W_{qk}^{l,h}$ as per Equation 1, the Fisher information of the corresponding mask $m_{mha}^{l,h}$ is taken for F_j. Including the value parameters here was found to empirically lead to a lower performance.

2. If the parameter in a particular row r of the intermediate layer of the feed-forward block, or $W_{mlp1}^{l,r*}$ as per Equation 2, the Fisher information of the corresponding mask $m_{mlp}^{l,r}$ is taken taken for F_j.

3. For all other parameters, $F_j = 1$, resulting in merging equivalent to plain averaging.

We note that our methodology requires the calculation of gradients for only $(H + D) \times L$ parameters, in comparison to Fisher-weighted averaging requiring $|\theta|$, almost certain to be magnitudes higher.

4. Evaluation

We evaluate our methodology on multiple pre-trained sizes of BERT, specifically tiny (Turc et al., 2019), base, and large (Devlin et al., 2019), along with the base model for the BERT derivative RoBERTa (Liu et al., 2019). We note however that with minimal modification it can be applied to all Transformer-based models.

We use publicly available checkpoints of all models finetuned on particular tasks in the GLUE benchmark (Wang et al., 2018): MNLI (Williams et al., 2018), SST-2 (Socher et al., 2013), MRPC (Dolan and Brockett, 2005), QQP (Iyer et al., 2017), QNLI (Rajpurkar et al., 2016), and RTE (Wang et al., 2009). While the datasets we use are wholly in English, no part of our methodology depends on the specific language. However, we assume that for two models being merged, the datasets they were trained on should be in the same language for valid results. We leave investigation of cross-lingual model merging to future work.

For each checkpoint, we take 128, 2,048, and 32,768 samples from the training set of the corresponding dataset to calculate the Fisher information. We then merge the models, two at a time,

Model (↓)	Averaging	Fisher (2022)			Ours		
No. of Samples (→)		128	2,048	32,768	128	2,048	32,768
BERT (Base)	91.2	92.0	91.7	91.9	**95.2** (+4.0)	92.8 (+1.6)	93.7 (+2.5)
BERT (Large)	85.8	87.0	86.1	89.2	**90.1** (+4.3)	88.1 (+2.3)	87.9 (+2.1)
BERT (Tiny)	**90.8**	89.6	89.2	90.4	86.1 (-4.7)	88.4 (-2.4)	88.7 (-2.1)
RoBERTa	86.2	91.5	88.5	88.5	**92.7** (+6.5)	89.4 (+3.2)	88.8 (+2.6)

Table 1: Normalized and aggregated metrics across tasks.

and evaluate on the validation set of both the corresponding datasets. As the final classification head layer for each finetuned model is different and task-specific, we use the respective head for each task when evaluating. For comparison, the same procedure is conducted using the original Fisher-weighted merging methodology in (Matena and Raffel, 2022), along with simple averaging (Wortsman et al., 2022; Choshen et al., 2022).

5. Results

Table 1 showcases the performance of our proposed method compared to simple averaging and the Fisher-weighted model merging of Matena and Raffel (2022). For ease of comparison, the difference between our method and the simple averaging is also indicated. Results are normalized based on the respective fine-tuned model's performance on each task, and the median is taken to reduce the effects of outliers. The results demonstrate notable variations based on the model size and the number of samples used for calculating Fisher information.

For BERT (Base), our method demonstrates a consistent improvement over other methods across all sample sizes, with the most significant gain seen at 128 samples, providing a **+4.0** increase in performance from simple averaging. BERT (Large) and RoBERTa follow a similar trend, with our method consistently outperforming the others. The greatest increase is again at 128 samples, with a performance boost of **+4.0** and **+6.5** respectively.

The results for BERT (Tiny) show an anomalous decrease in performance from simple averaging, in both Fisher-weighted merging and our method. However, we note that our method closes the gap as the number of samples increases, with a -2.1 decrease using 32,768 samples. We also stress the fact that BERT (Tiny) is an exceptionally small model, with only two transformer layers and a hidden embedding size of 128.

We note that our method also provides a significant efficiency advantage over Fisher-weighted averaging, with elapsed time speed-ups of **66.2x**, **57.4x**, **321.7x**, and **69.5x**, for BERT (Base), BERT (Large), BERT (Tiny), and RoBERTa respectively. To reduce the effect of overhead, the speedup is calculated using only the average runtime of the Fisher information approximation algorithm for each method, with 128 samples, running on a Nvidia T4 GPU.

Overall, these results confirm the efficacy of using the Fisher information of mask nodes for model merging, particularly in scenarios with limited data or computational resources. Further investigation is needed to fully understand the dynamics with scale, particularly for extremely small models like BERT (Tiny).

6. Conclusion

In this paper, we introduced a novel model merging method for Transformers that builds upon the Fisher-weighted averaging methodology. By calculating the Fisher information of masks in attention heads and feed-forward layers, our approach significantly reduces the computational cost associated with full-scale Fisher-weighted merging while improving performance. Without a dependence on the validation set, our method is widely applicable for model merging in the context of multi-task learning.

We evaluated our method across several models and tasks, demonstrating its effectiveness and efficiency. The low computation cost enables practical model merging for real-world applications, particularly in resource-constrained environments. Future work could be done to evaluate our method's capabilities for merging 3+ models at a time. Novel mask schemes may also be considered, along with an extension to different architectures.

7. Bibliographical References

Samuel K. Ainsworth, Jonathan Hayase, and Siddhartha Srinivasa. 2023. Git re-basin: Merging models modulo permutation symmetries.

Devansh Arpit, Huan Wang, Yingbo Zhou, and Caiming Xiong. 2022. Ensemble of averages: Improving model selection and boosting performance in domain generalization.

Junbum Cha, Sanghyuk Chun, Kyungjae Lee, Han-Cheol Cho, Seunghyun Park, Yunsung Lee, and Sungrae Park. 2021. Swad: Domain generalization by seeking flat minima.

Leshem Choshen, Elad Venezian, Noam Slonim, and Yoav Katz. 2022. Fusing finetuned models for better pretraining.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Rahim Entezari, Hanie Sedghi, Olga Saukh, and Behnam Neyshabur. 2022. The role of permutation invariance in linear mode connectivity of neural networks.

Chris Fifty, Ehsan Amid, Zhe Zhao, Tianhe Yu, Rohan Anil, and Chelsea Finn. 2021. Efficiently identifying task groupings for multi-task learning. *Advances in Neural Information Processing Systems*, 34:27503–27516.

Xu Han, Zhengyan Zhang, Ning Ding, Yuxian Gu, Xiao Liu, Yuqi Huo, Jiezhong Qiu, Yuan Yao, Ao Zhang, Liang Zhang, Wentao Han, Minlie Huang, Qin Jin, Yanyan Lan, Yang Liu, Zhiyuan Liu, Zhiwu Lu, Xipeng Qiu, Ruihua Song, Jie Tang, Ji-Rong Wen, Jinhui Yuan, Wayne Xin Zhao, and Jun Zhu. 2021. Pre-trained models: Past, present and future. *AI Open*, 2:225–250.

Awni Hannun, Chuan Guo, and Laurens van der Maaten. 2021. Measuring data leakage in machine-learning models with fisher information.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. 2023. Editing models with task arithmetic.

Xisen Jin, Xiang Ren, Daniel Preotiuc-Pietro, and Pengxiang Cheng. 2023. Dataless knowledge fusion by merging weights of language models.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2017a. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2017b. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526.

Woosuk Kwon, Sehoon Kim, Michael W. Mahoney, Joseph Hassoun, Kurt Keutzer, and Amir Gholami. 2022. A fast post-training pruning framework for transformers. In *Advances in Neural Information Processing Systems*.

Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. 2020. On the convergence of fedavg on non-iid data.

Yixuan Li, Jason Yosinski, Jeff Clune, Hod Lipson, and John Hopcroft. 2016. Convergent learning: Do different neural networks learn the same representations?

Liyang Liu, Shilong Zhang, Zhanghui Kuang, Aojun Zhou, Jing-Hao Xue, Xinjiang Wang, Yimin Chen, Wenming Yang, Qingmin Liao, and Wayne Zhang. 2021. Group fisher pruning for practical network compression. In *International Conference on Machine Learning*, pages 7021–7032. PMLR.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Michael Matena and Colin Raffel. 2022. Merging models with fisher-weighted averaging.

H. B. McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. 2016. Communication-efficient learning of deep networks from decentralized data. In *International Conference on Artificial Intelligence and Statistics*.

Bonan Min, Hayley Ross, Elior Sulem, Amir Pouran Ben Veyseh, Thien Huu Nguyen, Oscar Sainz, Eneko Agirre, Ilana Heinz, and Dan Roth. 2021. Recent advances in natural language processing via large pre-trained language models: A survey.

Razvan Pascanu and Yoshua Bengio. 2014. Revisiting natural gradient for deep networks.

Mary Phuong and Marcus Hutter. 2022. Formal algorithms for transformers.

XiPeng Qiu, TianXiang Sun, YiGe Xu, YunFan Shao, Ning Dai, and XuanJing Huang. 2020. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, 63(10):1872–1897.

Sidak Pal Singh and Martin Jaggi. 2023. Model fusion via optimal transport.

Alexander Soen and Ke Sun. 2021. On the variance of the fisher information for deep learning.

N. Joseph Tatro, Pin-Yu Chen, Payel Das, Igor Melnyk, Prasanna Sattigeri, and Rongjie Lai. 2020. Optimizing mode connectivity via neuron alignment.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: On the importance of pre-training compact models.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Mitchell Wortsman, Gabriel Ilharco, Samir Yitzhak Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S. Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. 2022. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. 2023. Resolving interference when merging models.

Yu Zhang and Qiang Yang. 2021. A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering*, 34(12):5586–5609.

Ce Zhou, Qian Li, Chen Li, Jun Yu, Yixin Liu, Guangjing Wang, Kai Zhang, Cheng Ji, Qiben Yan, Lifang He, Hao Peng, Jianxin Li, Jia Wu, Ziwei Liu, Pengtao Xie, Caiming Xiong, Jian Pei, Philip S. Yu, and Lichao Sun. 2023. A comprehensive survey on pretrained foundation models: A history from bert to chatgpt.

8. Language Resource References

Dolan, William B and Brockett, Chris. 2005. *Microsoft Research Paraphrase Corpus*. Microsoft. [link].

Shankar Iyer and Nikhil Dandekar and and Kornél Csernai. 2017. *Quora Question Pairs*. Quora. [link].

Rajpurkar, Pranav and Zhang, Jian and Lopyrev, Konstantin and Liang, Percy. 2016. *Stanford Question Answering Dataset*. Standford Natural Language Processing Group. Association for Computational Linguistics. [link].

Socher, Richard and Perelygin, Alex and Wu, Jean and Chuang, Jason and Manning, Christopher D and Ng, Andrew and Potts, Christopher. 2013. *Stanford Sentiment Treebank*. The Stanford Natural Language Processing Group. [link].

Wang, Alex and Singh, Amanpreet and Michael, Julian and Hill, Felix and Levy, Omer and Bowman, Samuel. 2009. *Recognizing Textual Entailment Dataset*. National Institute of Standards and Technology. [link].

Williams, Adina and Nangia, Nikita and Bowman, Samuel R. 2018. *Multi-Genre Natural Language Inference (MultiNLI) Corpus*. Courant Institute of Mathematical Sciences. [link].

Appendix

1. Checkpoints

For all architectures we mention, 6 models fine-tuned on the 6 respective tasks from the GLUE dataset were used. The checkpoints are given below.

1.1. BERT Base

- **MNLI**: `JeremiahZ/bert-base-uncased-mnli`
- **QQP**: `textattack/bert-base-uncased-QQP`
- **QNLI**: `textattack/bert-base-uncased-QNLI`
- **SST2**: `textattack/bert-base-uncased-SST-2`
- **MRPC**: `textattack/bert-base-uncased-MRPC`
- **RTE**: `textattack/bert-base-uncased-RTE`

1.2. BERT Large

- **MNLI**: `yoshitomo-matsubara/bert-large-uncased-mnli`
- **QQP**: `yoshitomo-matsubara/bert-large-uncased-qqp`
- **QNLI**: `yoshitomo-matsubara/bert-large-uncased-qnli`
- **SST2**: `yoshitomo-matsubara/bert-large-uncased-sst2`
- **MRPC**: `yoshitomo-matsubara/bert-large-uncased-mrpc`
- **RTE**: `yoshitomo-matsubara/bert-large-uncased-rte`

1.3. BERT Tiny

- **MNLI**: `M-FAC/bert-tiny-finetuned-mnli`
- **QQP**: `M-FAC/bert-tiny-finetuned-qqp`
- **QNLI**: `M-FAC/bert-tiny-finetuned-qnli`
- **SST2**: `M-FAC/bert-tiny-finetuned-sst2`
- **MRPC**: `M-FAC/bert-tiny-finetuned-mrpc`
- **RTE**: `muhtasham/bert-tiny-finetuned-glue-rte`

1.4. RoBERTa

- **MNLI**: `JeremiahZ/roberta-base-mnli`
- **QQP**: `JeremiahZ/roberta-base-qqp`
- **QNLI**: `JeremiahZ/roberta-base-qnli`
- **SST2**: `JeremiahZ/roberta-base-sst2`
- **MRPC**: `JeremiahZ/roberta-base-mrpc`
- **RTE**: `JeremiahZ/roberta-base-rte`

2. Compute and Efficiency

In order to compare the efficiency between our method and full Fisher-weighted merging (Matena and Raffel, 2022), we calculate the elapsed running times for both, for 128 samples on an Nvidia T4 GPU. As mentioned in Section 5, only the compute taken for the Fisher information approximation algorithm for both methods is considered: the merging itself is negligible in computation. The times for our method are given in Table 2 and for full Fisher calculation in Table 3.

We also calculate the FLOPS taken for our method with 128 samples, across all tasks and models, given in Table 4.

Task (↓)	**Ours, Time (s)**			
Model (→)	**Tiny**	**Base**	**Large**	**RoBERTa**
MNLI	0.056	1.121	4.591	1.195
QQP	0.063	0.670	2.405	0.630
QNLI	0.058	1.117	4.321	1.119
SST2	0.046	0.565	3.624	0.516
MRPC	0.048	0.972	3.624	0.970
RTE	0.053	1.251	5.047	1.307
Mean	**0.055**	**0.959**	**4.003**	**0.993**

Table 2: Time taken for our method with 128 samples across all tasks and models.

Task (↓)	**Fisher (2022), Time (s)**			
Model (→)	**Tiny**	**Base**	**Large**	**RoBERTa**
MNLI	19.149	71.486	284.276	73.203
QQP	17.472	55.193	201.505	56.623
QNLI	16.510	55.462	201.746	56.457
SST2	16.836	56.132	203.036	57.657
MRPC	16.376	55.973	202.999	57.304
RTE	17.011	55.851	202.527	57.856
Mean	**17.101**	**58.009**	**214.299**	**59.383**

Table 3: Time taken for full Fisher calculation with 128 samples across all tasks and models.

Task (↓)	**Ours, GFLOPS**			
Model (→)	**Tiny**	**Base**	**Large**	**RoBERTa**
MNLI	21.81	4,411.89	15,649.25	4,366.00
QQP	12.21	2,588.03	9,208.30	2,521.04
QNLI	25.08	5,008.74	17,750.74	5,054.74
SST2	9.25	1,988.07	7,080.03	1,976.64
MRPC	19.10	3,917.91	13,910.27	3,963.80
RTE	29.09	5,734.45	20,303.87	5,734.45
Mean	**18.64**	**4,172.11**	**14,692.66**	**4,138.64**

Table 4: FLOPS for our method with 128 samples across all tasks and models.